Reference Manual for libRASCH-0.7.4

Raphael Schneider

Reference Manual for libRASCH-0.7.4
by Raphael Schneider

libRASCH Reference Manual Version 0.1.0 Edition
Published 2004
Copyright © 2004 Raphael Schneider

Table of Contents

1. Function Reference for C 1

ra_lib_init 1
ra_lib_close 1
ra_lib_get_error 2
ra_lib_handle_from_any_handle 3
ra_meas_handle_from_any_handle 4
ra_lib_use_plugin 4
ra_plugin_get_by_num 5
ra_plugin_get_by_name 6
ra_meas_find_first 7
ra_meas_find_next 8
ra_meas_close_find 9
ra_meas_open 10
ra_meas_new 10
ra_meas_save 11
ra_meas_close 12
ra_info_get 13
ra_info_get_by_name 14
ra_info_get_by_idx 15
ra_info_set 16
ra_rec_get_first 17
ra_rec_get_next 17
ra_rec_get_first_child 18
ra_rec_add 19
ra_dev_add 20
ra_ch_add 20
ra_session_new 21
ra_raw_add 22
ra_raw_get 23
ra_raw_get_unit 24
ra_gui_call 25
ra_proc_get 26
ra_proc_free 27
ra_proc_do 27
ra_proc_get_result 28
ra_view_get 29
ra_lib_get_option 30
ra_lib_set_option 31
ra_meas_copy 32
ra_meas_move 32
ra_meas_delete 33
ra_value_malloc 34
ra_value_free 35
ra_value_reset 35
ra_value_get_type 36
ra_value_is_ok 37

ra_value_get_num_elem 38
ra_value_get_info 38
ra_value_get_name 39
ra_value_get_desc 40
ra_value_set_number 41
ra_value_get_number 41
ra_value_set_long 42
ra_value_set_double 43
ra_value_set_string 44
ra_value_set_string_utf8 45
ra_value_set_long_array 45
ra_value_set_double_array 46
ra_value_set_string_array 47
ra_value_set_string_array_utf8 48
ra_value_get_long 49
ra_value_get_double 50
ra_value_get_string 50
ra_value_get_string_utf8 51
ra_value_get_long_array 52
ra_value_get_double_array 53
ra_value_get_string_array 54
ra_value_get_string_array_utf8 54
ra_value_copy 55
ra_list_add 56
ra_list_insert 57
ra_list_del 58
ra_comm_add 59
ra_comm_del 60
ra_comm_emit 61
ra_eval_add 61
ra_eval_get 62
ra_eval_get_def 63
ra_eval_get_orig 64
ra_eval_delete 65
ra_eval_set_def 65
ra_ev_insert 66
ra_ev_del 67
ra_evset_get_num_events 68
ra_evset_set_start_end 69
ra_evset_get_start_end 70
ra_ev_get_value 71
ra_ev_set_value 72
ra_ev_get_value_inf 72
ra_ev_set_value_inf 73
ra_prop_add 74
ra_prop_add_name 75
ra_prop_add_custom 76
ra_prop_del 78

ra_prop_get_by_num ... 78
ra_prop_get_by_name ... 79
ra_evset_get_by_num ... 80
ra_evset_get_by_name ... 81
ra_evset_get_by_prop ... 82
ra_evset_get_flag_prop ... 82
ra_eval_loadxml ... 83
ra_eval_savexml ... 84
ra_templ_add ... 85
ra_templ_del ... 86
ra_templ_add_single ... 87
ra_templ_del_single ... 88
ra_templ_get_raw ... 89
2. Structure Reference for C ... 90
3. Function Reference for Perl (OO-Interface) ... 91
4. Function Reference for Python (OO-Interface) ... 92
5. Function Reference for Matlab/Octave ... 93
6. Infos Reference ... 94
7. Signals for Inter Plugin Communication ... 101
select-event ... 101
highlight-event ... 101
eval-change ... 102
start-event-change ... 103
end-event-change ... 103
event-change ... 104
region-change ... 105
add-event ... 105
del-event-begin ... 106
del-event-end ... 107
template-delete ... 108
template-add ... 108
pos-change ... 109
gui-event-start ... 110
gui-event-end ... 110
x-resolution ... 111
y-resolution ... 112
8. libRASCH Plugins ... 114
8.1. Access Plugins ... 114
ART ... 114
cfs ... 114
ctg-rasch ... 115
ctg-sonicaid ... 116
dadisp ... 116
dasylab ... 117
edf/edf+ ... 117
ishne-holter ... 118

mit-bih 119
portapres 119
poly5/tms32 120
read-rri 120
8.2. Process Plugins 121
ap-morphology 121
calibration 122
dawes-redman 124
detect-ctg 127
detect-simple 128
ecg 129
fiducial-point 134
freq-analysis 135
hrv 137
template 139
respiration 141
8.3. GUI/View Plugins 141
ch-select-dlg 142
eval-dlg 142
plugin-info-dlg 143
sig-sel-dlg 143
simple-calc-dlg 144
cont-ap-view 145
ctg-view 146
ecg-view 146
plot-view 147
rasch-view 147
template-view 148
ts-view 149

List of Tables

6-1. List of info-names for library infos -- ra_lib_get_info 94
6-2. List of info-names for measurement infos -- ra_meas_get_info 94
6-3. List of info-names for measurement-object (person) infos -- ra_obj_get_info 94
6-4. List of info-names for measurement-object (patient) infos -- ra_obj_get_info 95
6-5. List of info-names for recording infos -- ra_rec_get_info 96
6-6. List of info-names for device infos -- ra_dev_get_info 97
6-7. List of info-names for channel infos -- ra_ch_get_info 97
6-8. List of info-names for evaluation infos -- ra_eval_get_info 98
6-9. List of info-names for event-property infos -- ra_prop_get_info 98
6-10. List of info-names for template infos -- not yet implemented 99
6-11. List of info-names for single-template infos -- not yet implemented 99
6-12. List of info-names for plugin infos -- ra_plugin_get_info 99
6-13. List of info-names for plugin-results infos -- ra_proc_get_result_info 100
8-1. List of options for plugin ap-morphology 122
8-2. List of results of plugin ap-morphology 122
8-1. List of options for plugin calibration 123
8-2. List of results of plugin calibration 124
8-1. List of options for plugin dawes-redman 125
8-2. List of results of plugin dawes-redman 126
8-1. List of options for plugin detect-ctg 128
8-2. List of results of plugin detect-ctg 128
8-1. List of options for plugin detect-simple 129
8-2. List of results of plugin detect-simple 129
8-1. List of options for plugin ecg 130
8-2. List of results of plugin ecg 130
8-1. List of options for plugin fiducial-point 134
8-1. List of options for plugin freq-analysis 135
8-2. List of results of plugin freq-analysis 137
8-1. List of options for plugin hrv 138
8-2. List of results of plugin hrv 138
8-1. List of options for plugin template 140
8-1. List of options for plugin respiration 141
8-1. List of results of plugin ch-select-dlg 142
8-1. List of options for plugin sig-sel-dlg 144
8-2. List of results of plugin sig-sel-dlg 144

Chapter 1. Function Reference for C

No special intro text.

ra_lib_init

Name

`ra_lib_init` — init libRASCH

Synopsis

```
LIBRAAPI ra_handle ra_lib_init (void);
```

Arguments

None

Description

Init libRASCH, read config-file and init plugins; returns ra_handle.

source file

ra.c

ra_lib_close

Name

`ra_lib_close` — close libRASCH

Synopsis

```
LIBRAAPI void ra_lib_close (ra_handle h);
```

Arguments

h

handle to ra instance

Description

Close libRASCH, close plugins and frees all allocated memory.

source file

ra.c

ra_lib_get_error

Name

`ra_lib_get_error` — get last error

Synopsis

```
LIBRAAPI long ra_lib_get_error (ra_handle h, char * text, int len);
```

Arguments

h

handle to ra instance

text

character buffer which receives error text

len

size of character buffer text

Description

This function returns the last error occured in libRASCH (or in one of the plugins). See 'ra_error.h' for meaning of returned number. If the number is negative, the last error was an OS specific error and the error number comes from the OS (just remove the minus sign). If text is not NULL, a short error description will be returned in text.

source file

ra.c

ra_lib_handle_from_any_handle

Name

`ra_lib_handle_from_any_handle` — retrive handle of libRASCH instance from any handle

Synopsis

```
LIBRAAPI ra_handle ra_lib_handle_from_any_handle (any_handle h);
```

Arguments

h

handle

Description

Returns handle of libRASCH instance from any handle.

source file

ra.c

ra_meas_handle_from_any_handle

Name

`ra_meas_handle_from_any_handle` — retrive measurement-handle from any handle

Synopsis

```
LIBRAAPI meas_handle ra_meas_handle_from_any_handle (any_handle h);
```

Arguments

h

handle

Description

Returns measurement-handle.

source file

ra.c

ra_lib_use_plugin

Name

`ra_lib_use_plugin` — set using a plugin (or not)

Synopsis

```
LIBRAAPI int ra_lib_use_plugin (ra_handle h, int plugin_index, int use_it);
```

Arguments

h

handle to libRASCH

plugin_index

index of plugin to change

use_it

flag using plugin (1: use plugin, !1: do not use plugin)

Description

Use or do not use plugin.

source file

ra.c

ra_plugin_get_by_num

Name

`ra_plugin_get_by_num` — get plugin-handle

Synopsis

```
LIBRAAPI plugin_handle ra_plugin_get_by_num (ra_handle h, int plugin_num, int
search_all);
```

Arguments

h

handle to libRASCH

plugin_num

number of plugin

search_all

flag if all plugins should be searched (or only those with the use-it-flag enabled)

Description

Returns the plugin-handle for plugin #plugin_num.

source file

ra.c

ra_plugin_get_by_name

Name

ra_plugin_get_by_name — get plugin-handle

Synopsis

```
LIBRAAPI plugin_handle ra_plugin_get_by_name (ra_handle h, const char * name,
int search_all);
```

Arguments

h

handle to libRASCH

name

name of plugin

search_all

flag if all plugins should be searched (or only those with the use-it-flag enabled)

Description

Returns the plugin-handle for plugin with name name.

source file

ra.c

ra_meas_find_first

Name

ra_meas_find_first — find libRASCH-handled measurements (return first one)

Synopsis

```
LIBRAAPI ra_find_handle ra_meas_find_first (ra_handle h, const char * dir,
struct ra_find_struct * rfs);
```

Arguments

h

handle to libRASCH

dir

directory which should be scanned for measurements

rfs

pointer to ra_find_struct which will held the infos for the found measurement.

Description

Search for libRASCH-handled measurements in dir. If at least on measurement was found a valid (!NULL) ra_find_handle will be returned and the infos about the measurement is in mfs.

source file

ra.c

ra_meas_find_next

Name

`ra_meas_find_next` — find libRASCH-handled measurements (returns next)

Synopsis

```
LIBRAAPI int ra_meas_find_next (ra_find_handle h, struct ra_find_struct *
rfs);
```

Arguments

h

find-handle

rfs

pointer to ra_find_struct which will held the infos for the found measurement.

Description

Returns 1 if another measurement was found (the information will be in mfs) or 0 if there are no more measurements searched with ra_meas_find_first().

source file

ra.c

ra_meas_close_find

Name

`ra_meas_close_find` — close the search for libRASCH-handled measurements

Synopsis

```
LIBRAAPI void ra_meas_close_find (ra_find_handle h);
```

Arguments

h

find-handle

Description

End a search for libRASCH-handled measurements (frees some memory). Must be called if the ra_find_handle h returned from ra_meas_find_first() will no be longer used.

source file

ra.c

ra_meas_open

Name

`ra_meas_open` — open a measurement

Synopsis

```
LIBRAAPI meas_handle ra_meas_open (ra_handle h, const char * file, int fast);
```

Arguments

h

handle to libRASCH

file

file-name of the measurement

fast

flag if some init-code should be done (=0) or not (=1)

Description

Opens the measurement file and returns a handle to the measurement. If the fast-flag is set, some initialisation-code (e.g. reading evaluation infos from the files) will not be done. This can be useful if only information about the measurment-object is needed but not about the evaluation. Getting the evaluation infos can be "very" time consuming.

source file

ra.c

ra_meas_new

Name

ra_meas_new — creates a new measurement

Synopsis

```
LIBRAAPI meas_handle ra_meas_new (ra_handle h, const char * dir, const char *
name);
```

Arguments

h

handle to libRASCH

dir

directory where the measurement-files should be stored

name

name of the measurement

Description

Creates a new measurement with the name name in the directory dir. The function returns the handle to the measurement. The measurement will be stored using the libRASCH file-format. Not full implemented. !!!

source file

ra.c

ra_meas_save

Name

`ra_meas_save` — saves a measurement

Synopsis

```
LIBRAAPI int ra_meas_save (meas_handle mh);
```

Arguments

mh

handle to measurement

Description

Saves the changes to a measurement (NOT to the evaluation). Plugin must support this. Not full implemented. !!!

source file

ra.c

ra_meas_close

Name

`ra_meas_close` — close a measurement

Synopsis

```
LIBRAAPI void ra_meas_close (meas_handle mh);
```

Arguments

mh

handle to measurement

Description

Close a measurement.

source file

ra.c

ra_info_get

Name

`ra_info_get` — get infos

Synopsis

```
LIBRAAPI int ra_info_get (any_handle h, int id, value_handle vh);
```

Arguments

h

handle

id

id of information wanted

vh

value_handle receiving the info

Description

Returns 0 if the wanted information was found and != 0 if not. The information will be in vh. For usage of the function please see user-manual.

source file

ra.c

ra_info_get_by_name

Name

`ra_info_get_by_name` — get infos

Synopsis

```
LIBRAAPI int ra_info_get_by_name (any_handle h, const char * name,
value_handle vh);
```

Arguments

h

handle

name

name (text string) of information wanted

vh

value_handle receiving the info

Description

Returns 0 if the wanted information was found and != 0 if not. The information will be in vh. For usage of the function please see user-manual.

source file

ra.c

ra_info_get_by_idx

Name

`ra_info_get_by_idx` — get infos

Synopsis

```
LIBRAAPI int ra_info_get_by_idx (any_handle h, int info_type, int idx,
value_handle vh);
```

Arguments

h

measurement-handle

info_type

type of information wanted

idx

index of information wanted

vh

value_handle receiving the info

Description

Returns 0 if the wanted information was found and != 0 if not. The information will be in vh. For usage of the function please see user-manual.

source file

ra.c

ra_info_set

Name

`ra_info_set` — set info

Synopsis

```
LIBRAAPI int ra_info_set (any_handle h, value_handle vh);
```

Arguments

h

handle

vh

value_handle with the info

Description

Not implemented !!!

source file

ra.c

ra_rec_get_first

Name

`ra_rec_get_first` — get root-recording

Synopsis

```
LIBRAAPI rec_handle ra_rec_get_first (meas_handle mh, long session);
```

Arguments

mh

measurement-handle

session

session number

Description

Returns the first recording-handle of session session.

source file

ra.c

ra_rec_get_next

Name

ra_rec_get_next — get next recording

Synopsis

```
LIBRAAPI rec_handle ra_rec_get_next (rec_handle rh);
```

Arguments

rh

recording-handle

Description

Returns the next recording which follows recording rh.

source file

ra.c

ra_rec_get_first_child

Name

ra_rec_get_first_child — get first sub-recording

Synopsis

```
LIBRAAPI rec_handle ra_rec_get_first_child (rec_handle rh);
```

Arguments

rh

recording-handle

Description

Returns the first child-recording of recording rh.

source file

ra.c

ra_rec_add

Name

ra_rec_add — add a recording

Synopsis

```
LIBRAAPI rec_handle ra_rec_add (meas_handle mh, rec_handle parent);
```

Arguments

mh

handle of the measurement

parent

handle of the parent recording (or NULL)

Description

Adds a recording to a measurement. The recording will be a child-recording of the parent-recording parent. The recording-handle will be returned. Not implemented yet. !!!

source file

ra.c

ra_dev_add

Name

`ra_dev_add` — add a recording-device

Synopsis

```
LIBRAAPI int ra_dev_add (rec_handle rh);
```

Arguments

rh

handle of the recording

Description

Adds a device to the recording rh. The number of the device will be returned. Not implemented yet. !!!

source file

ra.c

ra_ch_add

Name

ra_ch_add — add a recording-channel

Synopsis

```
LIBRAAPI int ra_ch_add (rec_handle rh);
```

Arguments

rh

handle of the recording

Description

Adds a channel to the recording rh. The number of the channel will be returned. Not implemented yet. !!!

source file

ra.c

ra_session_new

Name

ra_session_new — start a new recording session

Synopsis

```
LIBRAAPI int ra_session_new (meas_handle mh);
```

Arguments

mh

handle of the measurement

Description

Starts a new recording session and close the previous one. The function returns the new session number. Not implemented yet. !!!

source file

ra.c

ra_raw_add

Name

`ra_raw_add` — add raw data

Synopsis

```
LIBRAAPI size_t ra_raw_add (meas_handle mh, unsigned int ch, value_handle
vh);
```

Arguments

mh

handle of the measurement

ch

channel where data should be added

vh

data to be added

Description

Adds raw data to a measurement. The data is added to the current session. Not implemented yet. !!!

source file

ra.c

ra_raw_get

Name

`ra_raw_get` — get raw-signal data

Synopsis

```
LIBRAAPI size_t ra_raw_get (rec_handle rh, unsigned int ch, size_t start,
size_t num_data, DWORD * data, DWORD * data_high);
```

Arguments

rh

recording-handle

ch

channel

start

first sample

num_data

number of samples

data

buffer for samples

data_high

buffer for higher 4 bytes of samples (if size of one sample > 4 bytes)

Description

This function returns the sample's (raw-signal data) from the measurement mh and the recording rh. The first sample in data will be the sample number start (in sampleunits) and max. num_data are in data. The memory for buffer data must be allocated in calling function. The samples returned in this function are the samples as stored in the file. If the size of one sample is greater than 4 bytes, the upper part of the samples are stored in the data_high buffer. If the size is not greater than 4 bytes, the buffer is not needed.

source file

ra.c

ra_raw_get_unit

Name

ra_raw_get_unit — get raw-signal data scaled to 'unit'-values

Synopsis

```
LIBRAAPI size_t ra_raw_get_unit (rec_handle rh, unsigned int ch, size_t
start, size_t num_data, double * data);
```

Arguments

rh

recording-handle

ch

channel

start

first sample

num_data

number of samples

data

buffer for samples

Description

This function returns the sample's (raw-signal data) from the measurement mh and the recording rh. The first sample in data will be the sample number start (in sampleunits) and max. num_data are in data. The memory for buffer data must be allocated in calling function. The values are scaled to the unit of the channel.

source file

ra.c

ra_gui_call

Name

`ra_gui_call` — shows GUI-element of a plugin (if avail.)

Synopsis

```
LIBRAAPI int ra_gui_call (any_handle h, plugin_handle pl);
```

Arguments

h

processing-handle or view-handle

pl

plugin-handle

Description

If the plugin pl provides a GUI-element, this function shows this GUI-element and transfer control to it. The h variable contains some information, needed by the plugin. For more information if a plugin provides a GUI-element and what type/information is needed, please see the plugin specific documentation.

source file

ra.c

ra_proc_get

Name

`ra_proc_get` — get a processing handle

Synopsis

```
LIBRAAPI proc_handle ra_proc_get (plugin_handle pl);
```

Arguments

pl

plugin-handle

Description

This function returns a proc_handle for the plugin pl. The proc_handle will be returned initialized.

source file

ra.c

ra_proc_free

Name

`ra_proc_free` — free a processing handle

Synopsis

```
LIBRAAPI void ra_proc_free (proc_handle proc);
```

Arguments

`proc`

processing-handle

Description

This function frees a processing-handle and all associated memory.

source file

ra.c

ra_proc_do

Name

`ra_proc_do` — process the signal

Synopsis

```
LIBRAAPI int ra_proc_do (proc_handle proc);
```

Arguments

proc

processing-handle

Description

This function calls the "processing" function of a process-plugin. For more information what information in proc is needed, please see the plugin specific documentation.

source file

ra.c

ra_proc_get_result

Name

`ra_proc_get_result` — get the processing results

Synopsis

```
LIBRAAPI int ra_proc_get_result (proc_handle proc, value_handle vh);
```

Arguments

proc

processing-handle

vh

value-handle

Description

This function returns in inf one result from a processing perfomed with ra_proc_do. Please check plugin specific documentation what results are available.

source file

ra.c

ra_view_get

Name

`ra_view_get` — create/show view of view-plugin

Synopsis

```
LIBRAAPI int ra_view_get (view_handle vh);
```

Arguments

vh

view-handle

Description

This function creates/shows the GUI-element/window provided by a view-plugin. Please check plugin specific documentation what additional information is needed.

source file

ra.c

ra_lib_get_option

Name

`ra_lib_get_option` — returns an option value

Synopsis

```
LIBRAAPI int ra_lib_get_option (any_handle h, const char * opt_name,
value_handle vh);
```

Arguments

h

return option from this object

opt_name

name of the option

vh

after function-call, contains option value

Description

The function tries to return the option opt_name from the object h. At the moment only options for process-plugins (h needs to be a process-handle) are supported.

source file

ra.c

ra_lib_set_option

Name

`ra_lib_set_option` — set an option

Synopsis

```
LIBRAAPI int ra_lib_set_option (any_handle h, const char * opt_name,
value_handle vh);
```

Arguments

h

option will be set in this object

opt_name

name of the option

vh

option value

Description

The function set the option opt_name in object h. At the moment only options for process-plugins (h needs to be a process-handle) are supported.

source file

ra.c

ra_meas_copy

Name

`ra_meas_copy` — copy measurement

Synopsis

```
LIBRAAPI int ra_meas_copy (meas_handle mh, const char * dest_dir);
```

Arguments

mh

handle of measurement which should be copied

dest_dir

destination directory

Description

The function copy the files of measurement mh to directory dest_dir. IMPORTANT!!! Function _not_ completely tested. Use at your own risk.

source file

ra.c

ra_meas_move

Name

`ra_meas_move` — move measurement

Synopsis

```
LIBRAAPI meas_handle ra_meas_move (meas_handle mh, const char * dest_dir);
```

Arguments

mh

handle of measurement which should be moved

dest_dir

destination directory

Description

The function move the files of measurement mh to directory dest_dir. After the move the measurement will be reopend (to be able to handle the new positions of the files) and the functions returns the new measurement-handle. IMPORTANT!!! Function _not_ completely tested. Use at your own risk.

source file

ra.c

ra_meas_delete

Name

`ra_meas_delete` — delete measurement

Synopsis

```
LIBRAAPI int ra_meas_delete (meas_handle mh);
```

Arguments

mh

handle of measurement which should be deleted

Description

The function deletes the files of measurement mh. The measurement mh will be closed after the deletion. Therefore mh is no longer a valid measurement-handle. IMPORTANT!!! Function _not_ completely tested. Use at your own risk.

source file

ra.c

ra_value_malloc

Name

`ra_value_malloc` — allocate and initialize a value_handle

Synopsis

```
LIBRAAPI value_handle ra_value_malloc (void);
```

Arguments

None

Description

Returns a initialzed value-handle.

source file

value.c

ra_value_free

Name

`ra_value_free` — free value-handle

Synopsis

```
LIBRAAPI void ra_value_free (value_handle vh);
```

Arguments

vh

value-handle

Description

Frees the memory asocciated with vh.

source file

value.c

ra_value_reset

Name

`ra_value_reset` — reset value-handle

Synopsis

```
LIBRAAPI void ra_value_reset (value_handle vh);
```

Arguments

vh

value-handle

Description

Reset value-handle vh.

source file

value.c

ra_value_get_type

Name

`ra_value_get_type` — return value-type

Synopsis

```
LIBRAAPI long ra_value_get_type (value_handle vh);
```

Arguments

vh

value-handle

Description

Return the type of the value stored in vh. Check in ra_defines.h for the meaning of the returned value (RA_VALUE_TYPE_*).

source file

value.c

ra_value_is_ok

Name

`ra_value_is_ok` — checks if value is valid

Synopsis

```
LIBRAAPI int ra_value_is_ok (value_handle vh);
```

Arguments

vh

value-handle

Description

Returns ’1’ if value stored in vh is valid, ’0’ if not.

source file

value.c

ra_value_get_num_elem

Name

`ra_value_get_num_elem` — return number of elements of array values

Synopsis

```
LIBRAAPI long ra_value_get_num_elem (value_handle vh);
```

Arguments

vh

value-handle

Description

When an array is stored in vh, the number of elements of the array is returned.

source file

value.c

ra_value_get_info

Name

`ra_value_get_info` — returns id of info

Synopsis

```
LIBRAAPI long ra_value_get_info (value_handle vh);
```

Arguments

vh

value-handle

Description

Returns the id of the info set in vh. Check in ra_defines.h for for the meaning of info-id (RA_INFO_*).

source file

value.c

ra_value_get_name

Name

`ra_value_get_name` — returns name of info

Synopsis

```
LIBRAAPI const char * ra_value_get_name (value_handle vh);
```

Arguments

vh

value-handle

Description

Returns a character pointer to the name of the info set in vh. The pointer is valid until vh is free'd or a new info was retrived.

source file

value.c

ra_value_get_desc

Name

`ra_value_get_desc` — returns description of info

Synopsis

```
LIBRAAPI const char * ra_value_get_desc (value_handle vh);
```

Arguments

vh

value-handle

Description

Returns a character pointer to the description of the info set in vh. The pointer is valid until vh is free'd or a new info was retrived.

source file

value.c

ra_value_set_number

Name

`ra_value_set_number` — set a number in a value-handle

Synopsis

```
LIBRAAPI int ra_value_set_number (value_handle vh, long number);
```

Arguments

vh

value-handle

number

number

Description

Set the number number in the value-handle vh. This number is used when infos about channels, recording devices and plugins. number is also used when processing results are retrived.

source file

value.c

ra_value_get_number

Name

ra_value_get_number — get a number set in a value-handle

Synopsis

```
LIBRAAPI long ra_value_get_number (value_handle vh);
```

Arguments

vh

value-handle

Description

Returns the number number set in the value-handle vh.

source file

value.c

ra_value_set_long

Name

ra_value_set_long — set long value in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_long (value_handle vh, const long value);
```

Arguments

vh

value-handle

value

value to set in ra_info

Description

This function set the value in vh.

source file

value.c

ra_value_set_double

Name

`ra_value_set_double` — set double value in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_double (value_handle vh, const double value);
```

Arguments

vh

value-handle

value

value to set in ra_info

Description

This function set the value in vh.

source file

value.c

ra_value_set_string

Name

`ra_value_set_string` — set a string in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_string (value_handle vh, const char * string);
```

Arguments

vh

value-handle

string

string to set in ra_info

Description

This function set the string (‘\\0‘-ended char *) in vh.

source file

value.c

ra_value_set_string_utf8

Name

`ra_value_set_string_utf8` — set a UTF-8 encoded string in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_string_utf8 (value_handle vh, const char *
string);
```

Arguments

vh

value-handle

string

string to set in ra_info

Description

This function set the string (‘\\0‘-ended char *) in vh.

source file

value.c

ra_value_set_long_array

Name

ra_value_set_long_array — set long-array values in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_long_array (value_handle vh, const long * array,
long num);
```

Arguments

vh

value-handle

array

array to set in ra_info

num

number of array elements

Description

This function set the long array array with num elements in vh.

source file

value.c

ra_value_set_double_array

Name

ra_value_set_double_array — set double-array values in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_double_array (value_handle vh, const double *
array, long num);
```

Arguments

vh

value-handle

array

array to set in ra_info

num

number of array elements

Description

This function set the double array array with num elements in vh.

source file

value.c

ra_value_set_string_array

Name

ra_value_set_string_array — set a string-array in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_string_array (value_handle vh, const char **
*array, long num);
```

Arguments

vh

value-handle

**array*

num

number of array elements

Description

This function set the string-array array with num elements in vh.

source file

value.c

ra_value_set_string_array_utf8

Name

ra_value_set_string_array_utf8 — set a string-array (UTF-8 encoded) in value-handle

Synopsis

```
LIBRAAPI void ra_value_set_string_array_utf8 (value_handle vh, const char **
*array, long num);
```

Arguments

vh

value-handle

**array*

num

number of array elements

Description

This function set the string-array ’array’ with ’num’ elements in vh. The strings in ’array’ are expected to be UTF-8 encoded.

source file

value.c

ra_value_get_long

Name

`ra_value_get_long` — return long value from value-handle

Synopsis

```
LIBRAAPI long ra_value_get_long (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns the long value set in vh.

source file

value.c

ra_value_get_double

Name

`ra_value_get_double` — return double value from value-handle

Synopsis

```
LIBRAAPI double ra_value_get_double (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns the double value set in vh.

source file

value.c

ra_value_get_string

Name

`ra_value_get_string` — return character pointer from value-handle

Synopsis

```
LIBRAAPI const char * ra_value_get_string (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a character pointer to the string set in vh. The pointer is valid until vh is used in another function or vh is freed.

source file

value.c

ra_value_get_string_utf8

Name

`ra_value_get_string_utf8` — return character pointer from value-handle (UTF-8 encoded)

Synopsis

```
LIBRAAPI const char * ra_value_get_string_utf8 (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a character pointer to the string set in vh. The string is UTF-8 encoded. The pointer is valid until vh is used in another function or vh is freed.

source file

value.c

ra_value_get_long_array

Name

`ra_value_get_long_array` — return pointer to long array from value-handle

Synopsis

```
LIBRAAPI const long * ra_value_get_long_array (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a pointer to the long array stored in vh. The pointer is valid until vh is used in another function or vh is freed. The number of elements of the array can be get by calling ra_value_get_num_elem().

source file

value.c

ra_value_get_double_array

Name

`ra_value_get_double_array` — return pointer to double array from value-handle

Synopsis

```
LIBRAAPI const double * ra_value_get_double_array (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a pointer to the double array stored in vh. The pointer is valid until vh is used in another function or vh is freed. The number of elements of the array can be get by calling ra_value_get_num_elem().

source file

value.c

ra_value_get_string_array

Name

`ra_value_get_string_array` — return pointer to character pointer array from value-handle

Synopsis

```
LIBRAAPI const char ** ra_value_get_string_array (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a pointer to the character pointer array stored in vh. The pointer is valid until vh is used in another function or vh is freed. The number of elements of the array can be get by calling ra_value_get_num_elem().

source file

value.c

ra_value_get_string_array_utf8

Name

ra_value_get_string_array_utf8 — return pointer to character pointer array from value-handle (UTF-8 encoded)

Synopsis

```
LIBRAAPI const char ** ra_value_get_string_array_utf8 (value_handle vh);
```

Arguments

vh

value-handle

Description

This function returns a pointer to the character pointer array stored in vh. The strings are UTF-8 encoded. The pointer is valid until vh is used in another function or vh is freed. The number of elements of the array can be get by calling ra_value_get_num_elem().

source file

value.c

ra_value_copy

Name

ra_value_copy — copy one value_handle to another value_handle

Synopsis

```
LIBRAAPI int ra_value_copy (value_handle dest, value_handle src);
```

Arguments

dest

destination value-handle

src

source value-handle

Description

This function copies the values from one value-handle to another value-handle and the name and description. The remaining fields are not copied.

source file

value.c

ra_list_add

Name

ra_list_add — add an entry to a linked list

Synopsis

```
int ra_list_add (void ** *head, void * item);
```

Arguments

**head*

item

item to be added

Description

This function adds ’item’ to the linked list identified by ’head’ (head of list).

source file

linked_list.c

ra_list_insert

Name

`ra_list_insert` — insert an entry to a linked list

Synopsis

```
int ra_list_insert (void ** *head, void * prev, void * item);
```

Arguments

**head*

prev

previous item

item

item to be added

Description

This function inserts ’item’ in a linked list (identified by ’head’). The item will come directly after ’prev’.

source file

linked_list.c

ra_list_del

Name

`ra_list_del` — delete an entry from a linked list

Synopsis

```
int ra_list_del (void ** *head, void * item);
```

Arguments

**head*

item

item to be deleted

Description

This function deletes ’item’ from a linked list (identified by ’head’).

source file

linked_list.c

ra_comm_add

Name

`ra_comm_add` — add a signal receiver

Synopsis

```
LIBRAAPI int ra_comm_add (meas_handle mh, plugin_handle p, dest_handle dh,
const char * sig_name);
```

Arguments

mh

measurement handle

p

plugin handle

dh

destination handle

sig_name

name of the signal

Description

This function adds a new signal receiver p and dh for the signal sig_name to the inter-plugin-communication. Whenever the signal sig_name is emitted, the signal will be send to the plugin p. The plugin-instance will be identified with the help of dh.

source file

pl_comm.c

ra_comm_del

Name

`ra_comm_del` — delete a signal receiver

Synopsis

```
LIBRAAPI void ra_comm_del (meas_handle mh, dest_handle dh, const char *
sig_name);
```

Arguments

mh

measurement handle

dh

destination handle

sig_name

name of the signal

Description

This function deletes the signal receiver dh for the signal sig_name.

source file

pl_comm.c

ra_comm_emit

Name

ra_comm_emit — emits a signal

Synopsis

```
LIBRAAPI void ra_comm_emit (meas_handle mh, const char * sender, const char *
sig_name, int num_para, struct comm_para * para);
```

Arguments

mh

measurement handle

sender

name of the sender (needed for debug, can be NULL)

sig_name

name of the signal

num_para

number of signal parameters

para

signal parameters

Description

This function emits the signal sig_name wit num_para parameters.

source file

pl_comm.c

ra_eval_add

Name

ra_eval_add — adds an evaluation

Synopsis

```
LIBRAAPI eval_handle ra_eval_add (meas_handle mh, const char * name, const
char * comment, int orig, int def);
```

Arguments

mh

measurement-handle

name

name of the evaluation

comment

comment of the evaluation

orig

flag if this evalution was performed with the measurement-specific system

def

flag if this evaluation should be the default evaluation.

Description

This function adds an evaluation to a measurement.

source file

eval.c

ra_eval_get

Name

`ra_eval_get` — returns an evaluation-handle

Synopsis

```
LIBRAAPI eval_handle ra_eval_get (meas_handle mh, int eval);
```

Arguments

mh

measurement-handle

eval

zero based index of the evaluation which handle should be returned

Description

This function returns the evaluation-handle from handle number eval.

source file

eval.c

ra_eval_get_def

Name

`ra_eval_get_def` — returns the default evaluation-handle

Synopsis

```
LIBRAAPI eval_handle ra_eval_get_def (meas_handle mh);
```

Arguments

mh

measurement-handle

Description

This function returns the default evaluation-handle.

source file

eval.c

ra_eval_get_orig

Name

`ra_eval_get_orig` — returns the original evaluation-handle

Synopsis

```
LIBRAAPI eval_handle ra_eval_get_orig (meas_handle mh);
```

Arguments

mh

measurement-handle

Description

This function returns the original evaluation-handle.

source file

eval.c

ra_eval_delete

Name

`ra_eval_delete` — delete an evaluation

Synopsis

```
LIBRAAPI int ra_eval_delete (eval_handle eh);
```

Arguments

eh

evaluation-handle

Description

This function deletes the evaluation eh.

source file

eval.c

ra_eval_set_def

Name

`ra_eval_set_def` — set an evaluation to the default evaluation

Synopsis

```
LIBRAAPI int ra_eval_set_def (eval_handle eh);
```

Arguments

eh

evaluation-handle

Description

This function sets the evaluation eh to the default evaluation.

source file

eval.c

ra_ev_insert

Name

`ra_ev_insert` — insert an event

Synopsis

```
LIBRAAPI long ra_ev_insert (evset_handle seth, long pos, long start, long
end);
```

Arguments

seth

event-set handle

pos

position of event

start

start of event

end

end of event

Description

This function adds the event on position pos to an event-set evsh. If it is a continues event, the start- and end-position (start and end) will be set also.

source file

event.c

ra_ev_del

Name

ra_ev_del — delete an event

Synopsis

```
LIBRAAPI void ra_ev_del (evset_handle seth, long event);
```

Arguments

seth

event-set handle

event

number of event

Description

This function deletes the event number event from the event-set evsh.

source file

event.c

ra_evset_get_num_events

Name

ra_evset_get_num_events — returns number of events

Synopsis

```
LIBRAAPI long ra_evset_get_num_events (evset_handle seth);
```

Arguments

seth

event-set handle

Description

This function returns the number of events in the event-set evsh.

source file

event.c

ra_evset_set_start_end

Name

`ra_evset_set_start_end` — set start- and end-position of an event

Synopsis

```
LIBRAAPI int ra_evset_set_start_end (evset_handle seth, long event, long
start, long end);
```

Arguments

`seth`

event-set handle

`event`

number of the event

`start`

new start-position of the event

`end`

new end-position of the event

Description

This function set the start- and end-position of the event number event.

source file

event.c

ra_evset_get_start_end

Name

`ra_evset_get_start_end` — get start- and end-position of an event

Synopsis

```
LIBRAAPI int ra_evset_get_start_end (evset_handle seth, long event, long *
start, long * end);
```

Arguments

seth

event-set handle

event

number of the event

start

start-position of the event

end

end-position of the event

Description

This function get the start- and end-position of the event number event. The postions will be stored in start and end.

source file

event.c

ra_ev_get_value

Name

`ra_ev_get_value` — returns the property of an event

Synopsis

```
LIBRAAPI double ra_ev_get_value (prop_handle proph, long event);
```

Arguments

`proph`

event-property handle

`event`

number of the event

Description

This function returns the event-property proph of the event event. At the moment it returns -1 if an error occured. The function has to be extended to use the libRASCH error-variable.

source file

event.c

ra_ev_set_value

Name

`ra_ev_set_value` — set the property of an event

Synopsis

```
LIBRAAPI int ra_ev_set_value (prop_handle proph, long event, double value);
```

Arguments

`proph`

event-property handle

`event`

number of the event

`value`

property-value

Description

This function sets the event-property proph of the event event to value.

source file

event.c

ra_ev_get_value_inf

Name

`ra_ev_get_value_inf` — returns the property of an event in a ra_info struct

Synopsis

```
LIBRAAPI int ra_ev_get_value_inf (prop_handle proph, long event, value_handle
vh, value_handle ch);
```

Arguments

proph

event-property handle

event

number of the event

vh

value asked for

ch

channel value (???)

Description

This function returns the event-property proph of the event event. If an error occured, -1 is returned. The value asked for is returned in inf.

source file

event.c

ra_ev_set_value_inf

Name

`ra_ev_set_value_inf` — set the property of an event to the value given in a ra_info struct

Synopsis

```
LIBRAAPI int ra_ev_set_value_inf (prop_handle proph, long event, value_handle
vh, value_handle ch);
```

Arguments

`proph`

event-property handle

`event`

number of the event

`vh`

value

`ch`

channel value (???)

Description

This function sets the event-property proph of the event event to the value given in the ra_info struct inf.

source file

event.c

ra_prop_add

Name

ra_prop_add — add a predefined event-property

Synopsis

```
LIBRAAPI prop_handle ra_prop_add (eval_handle eh, int event_prop);
```

Arguments

eh

evaluation handle

event_prop

number of event-property

Description

This function adds the predefined event-property number event_prop to the evaluation eh.

source file

event_prop.c

ra_prop_add_name

Name

ra_prop_add_name — add a predefined event-property (selected by name)

Synopsis

```
prop_handle ra_prop_add_name (eval_handle eh, const char * event_prop);
```

Arguments

eh

evaluation handle

event_prop

name of event-property

Description

This function adds the predefined event-property named event_prop to the evaluation eh.

source file

event_prop.c

ra_prop_add_custom

Name

ra_prop_add_custom — add an event-property

Synopsis

```
LIBRAAPI prop_handle ra_prop_add_custom (eval_handle eh, const char *
set_id_ascii, const char * set_name, const char * set_desc, int event_type,
const char * prop_id_ascii, const char * prop_name, const char * prop_desc,
int val_type, int use_minmax, double min, double max, const char * unit);
```

Arguments

eh

evaluation handle

set_id_ascii

event-set id-string (only ASCII char’s allowed)

set_name

event-set name

set_desc

event-set description

event_type

type of event (discrete or continues)

prop_id_ascii

event-property id-string (only ASCII char’s allowed)

prop_name

event-property name

prop_desc

event-property description

val_type

type of value (short, long or double)

use_minmax

are the following parameters min and max are valid

min

normal min. value of the property

max

normal max. value of the property

unit

unit of the property

Description

This function adds a new event-property the evaluation eh.

source file

event_prop.c

ra_prop_del

Name

`ra_prop_del` — delete an event-property from an evaluation

Synopsis

```
LIBRAAPI void ra_prop_del (prop_handle proph);
```

Arguments

proph

event-property handle

Description

Delete event property proph.

source file

event_prop.c

ra_prop_get_by_num

Name

ra_prop_get_by_num — returns an event-property handle

Synopsis

```
LIBRAAPI prop_handle ra_prop_get_by_num (eval_handle eh, int num);
```

Arguments

eh

evaluation handle

num

number of the event-property

Description

This function returns the event-property handle to the property number num.

source file

event_prop.c

ra_prop_get_by_name

Name

ra_prop_get_by_name — returns an event-property handle

Synopsis

```
LIBRAAPI prop_handle ra_prop_get_by_name (eval_handle eh, const char * name);
```

Arguments

eh

evaluation handle

name

name of the event-property

Description

This function returns the event-property handle to the property with name name.

source file

event_prop.c

ra_evset_get_by_num

Name

`ra_evset_get_by_num` — returns an event-set handle

Synopsis

```
LIBRAAPI evset_handle ra_evset_get_by_num (eval_handle eh, long num);
```

Arguments

eh

evaluation handle

num

number of the event-set

Description

This function returns the event-set handle to the set number num.

source file

event_prop.c

ra_evset_get_by_name

Name

`ra_evset_get_by_name` — returns an event-set handle

Synopsis

```
LIBRAAPI evset_handle ra_evset_get_by_name (eval_handle eh, const char *
name);
```

Arguments

eh

evaluation handle

name

name of the event-set

Description

This function returns the event-set handle to the set with name name.

source file

event_prop.c

ra_evset_get_by_prop

Name

`ra_evset_get_by_prop` — returns an event-set handle

Synopsis

```
LIBRAAPI evset_handle ra_evset_get_by_prop (prop_handle proph);
```

Arguments

proph

event-property handle

Description

This function returns the event-set handle which the event-property proph belongs.

source file

event_prop.c

ra_evset_get_flag_prop

Name

ra_evset_get_flag_prop — returns the event-property handling the event-set flags

Synopsis

```
LIBRAAPI prop_handle ra_evset_get_flag_prop (evset_handle seth);
```

Arguments

seth

event-set handle

Description

This function returns the event-property, which handles the event-flags associated with the event-set seth.

source file

event_prop.c

ra_eval_loadxml

Name

ra_eval_loadxml — load evalution in XML format

Synopsis

```
LIBRAAPI int ra_eval_loadxml (meas_handle mh, const char * file);
```

Arguments

mh

measurement handle

file

name of the evaluation file (optional)

Description

This function loads the evaluation(s) that belongs to the measurement mh. If no filename file is given, the default-filename will be used.

source file

load_xml.c

ra_eval_savexml

Name

`ra_eval_savexml` — save evalution in XML format

Synopsis

```
LIBRAAPI int ra_eval_savexml (meas_handle mh, const char * file, int
use_ascii);
```

Arguments

mh

measurement handle

file

name of the evaluation file (optional)

use_ascii

flag if event-values should be stored as ASCII-text (=1) or as

Description

binary MIME64 encoded data (=0) This function saves the evaluation(s) that belongs to the measurement mh. If no filename file is given, the default-filename will be used. If use_ascii is !=0 than the event values are stored as ASCII text.

source file

save_xml.c

ra_templ_add

Name

`ra_templ_add` — adds a template

Synopsis

```
LIBRAAPI int ra_templ_add (prop_handle proph, int num_ch, int * ch, int *
templ_size);
```

Arguments

proph

event-property-handle

num_ch

number of channels of the template

ch

channels used for the template

templ_size

size of the template for each channel

Description

This function set the basic template-information in proph.

source file

template.c

ra_templ_del

Name

ra_templ_del — deletes all templates

Synopsis

```
LIBRAAPI void ra_templ_del (prop_handle proph);
```

Arguments

proph

event-property-handle

Description

This function deletes all templates from proph.

source file

template.c

ra_templ_add_single

Name

`ra_templ_add_single` — adds a template

Synopsis

```
LIBRAAPI int ra_templ_add_single (prop_handle proph, int fiducial_offset,
long type, double ** *data, long templ_num);
```

Arguments

`proph`

event-property-handle

`fiducial_offset`

position of the fiducial-point in su from template-start

`type`

??

`*data`

`templ_num`

number of the template

Description

This function add a specific template and returns the zero-based index of the template.

source file

template.c

ra_templ_del_single

Name

`ra_templ_del_single` — deletes a template

Synopsis

```
LIBRAAPI int ra_templ_del_single (prop_handle proph, int num, int
del_templ_events);
```

Arguments

proph

event-property-handle

num

number of template which should be deleted

del_templ_events

flag if the events, associated with the template, should be deleted

Description

This function deletes one template from proph. If del_templ_events is set (!= 0), the events are deleted also.

source file

template.c

ra_templ_get_raw

Name

ra_templ_get_raw — returns template raw-data

Synopsis

```
LIBRAAPI int ra_templ_get_raw (prop_handle proph, int templ_num, int ch,
double * data);
```

Arguments

proph

event-property-handle

templ_num

number of template which should be deleted

ch

channel for which the data should be returned

data

buffer which receivs the template data (must be allocated by calling function)

Description

This function returns in data the raw-data of the template templ_num, from channel ch.

source file

template.c

Chapter 2. Structure Reference for C

No special intro text.

Just a dummy text.

Chapter 3. Function Reference for Perl (OO-Interface)

Nothing special.

Just a dummy text.

Chapter 4. Function Reference for Python (OO-Interface)

No special intro.

Just a dummy text.

Chapter 5. Function Reference for Matlab/Octave

No special intro.

Just a dummy text.

Chapter 6. Infos Reference

The tables in this section list the informations, which are available from libRASCH. For measurements, recordings and measurement objects not all informations must be available.

The first column (’Info Name’) list the info-name which must be used in the Perl, Python, Matlab, Octave functions to get informations. The second column (’Info Constant’) list the constants used in C/C++ programs to get informations. The third column (’Info Description’) gives a short description of the information returned from libRASCH.

Table 6-1. List of info-names for library infos -- ra_lib_get_info

Info Name	Info Constant	Info Description
num_plugins	RA_INFO_NUM_PLUGINS_L	number of plugins
lib_version	RA_INFO_VERSION_C	version of libRASCH

Table 6-2. List of info-names for measurement infos -- ra_meas_get_info

Info Name	Info Constant	Info Description
num_sessions	RA_INFO_NUM_SESSIONS_L	number of sessions in a measurement
num_obj_infos	RA_INFO_NUM_OBJ_INFOS_L	number of infos about a measurment object
num_rec_gen_infos	RA_INFO_NUM_REC_GEN_INFOS_L	number of general infos
num_rec_dev_infos	RA_INFO_NUM_REC_DEV_INFOS_L	number of infos about a recording device
num_rec_ch_infos	RA_INFO_NUM_REC_CH_INFOS_L	number of infos about a channel
num_eval_infos	RA_INFO_NUM_EVAL_INFOS_L	number of infos about an evaluation
num_eval	RA_INFO_EVAL_NUM_L	number of evaluations
max_samplerate	RA_INFO_MAX_SAMPLERATE_D	maximum samplerate used in the recording
ch_xscale	RA_INFO_CH_XSCALE_D	position scale-factor for a channel
meas_size	RA_INFO_SIZE_L	size needed on disc of measurement in Bytes
meas_file	RA_INFO_FILES_CA	files belonging to measurement
meas_path	RA_INFO_PATH_C	full path of measurement incl. measurement name
meas_in_dir	RA_INFO_DIR_L	flag if measurement was saved in directory

Table 6-3. List of info-names for measurement-object (person) infos -- ra_obj_get_info

Info Name	Info Constant	Info Description
person_name	RA_INFO_OBJ_PERSON_NAME_C	last name of the measurement object
person_forename	RA_INFO_OBJ_PERSON_FORENAME_C	first name of the measurement object
person_birthday	RA_INFO_OBJ_PERSON_BIRTHDAY_C	birthday of the measurement object
person_height	RA_INFO_OBJ_PERSON_HEIGHT_C	height of the measurement object
person_weight	RA_INFO_OBJ_PERSON_WEIGHT_C	weight of the measurement object
person_street	RA_INFO_OBJ_PERSON_STREET_C	street of the address of the measurement object
person_city	RA_INFO_OBJ_PERSON_CITY_C	city of the address the measurement object
person_postalcode	RA_INFO_OBJ_PERSON_POSTALCODE_C	postalcode of the address the measurement object
person_country	RA_INFO_OBJ_PERSON_COUNTRY_C	country of the address the measurement object
person_address	RA_INFO_OBJ_PERSON_ADDRESS_C	address the measurement object (sometimes this field contains the whole address)
person_phone1	RA_INFO_OBJ_PERSON_PHONE1_C	phone-number of the measurement object
person_phone2	RA_INFO_OBJ_PERSON_PHONE2_C	phone-number 2 of the measurement object
person_fax	RA_INFO_OBJ_PERSON_FAX_C	fax-number of the measurement object
person_email	RA_INFO_OBJ_PERSON_EMAIL_C	email-address of the measurement object
person_website	RA_INFO_OBJ_PERSON_WEBSITE_C	website of the measurement object
person_comment	RA_INFO_OBJ_PERSON_COMMENT_C	comment about the measurement object

Table 6-4. List of info-names for measurement-object (patient) infos -- ra_obj_get_info

Info Name	Info Constant	Info Description
patient_id	RA_INFO_OBJ_PATIENT_ID_C	ID of the patient
patient_reason	RA_INFO_OBJ_PATIENT_REASON_C	reason of the recording

Info Name	Info Constant	Info Description
patient_diagnosis	RA_INFO_OBJ_PATIENT_DIAGNOSIS_C	diagnosis
patient_therapy	RA_INFO_OBJ_PATIENT_THERAPY_C	therapy
patient_medication	RA_INFO_OBJ_PATIENT_MEDICATION_C	medication of the patient
patient_hospital	RA_INFO_OBJ_PATIENT_HOSPITAL_C	Hospital where the recording was performed
patient_department	RA_INFO_OBJ_PATIENT_DEPARTMENT_C	Department where the recording was performed
patient_doctor	RA_INFO_OBJ_PATIENT_DOCTOR_C	Doctor who analyze the evaluation of the recording
patient_examiner	RA_INFO_OBJ_PATIENT_EXAMINER_C	Person who evaluate the recording
preg_num_fetus	RA_INFO_OBJ_PREG_WOMAN_NUM_FETUS_L	number fetus
preg_gestation_date	RA_INFO_OBJ_PREG_WOMAN_GESTATION_DATE_C	gestation date

Table 6-5. List of info-names for recording infos -- ra_rec_get_info

Info Name	Info Constant	Info Description
rec_type	RA_INFO_REC_GEN_TYPE_L	type of recording (see RA_REC_TYPE_*)
rec_name	RA_INFO_REC_GEN_NAME_C	name of recording
rec_desc	RA_INFO_REC_GEN_DESC_C	description of the recording
rec_comment	RA_INFO_REC_GEN_COMMENT_C	comment about the recording
rec_time	RA_INFO_REC_GEN_TIME_C	start time of recording (hh:mm:ss)
rec_date	RA_INFO_REC_GEN_DATE_C	start date of recording (dd.mm.yyyy)
rec_duration_sec	RA_INFO_REC_GEN_DURATION_SEC_D	duration of the recording in seconds
rec_number	RA_INFO_REC_GEN_NUMBER_L	number of recording (0: main recording)
rec_num_sub_rec	RA_INFO_REC_GEN_NUM_SUB_REC_L	number of sub-recordings
rec_num_devices	RA_INFO_REC_GEN_NUM_DEVICES_L	number of devices used for recording
rec_num_channel	RA_INFO_REC_GEN_NUM_CHANNEL_L	number of channels

Info Name	Info Constant	Info Description
rec_path	RA_INFO_REC_GEN_PATH_C	full path of recording incl. recording name
rec_is_directory	RA_INFO_REC_GEN_DIR_L	flag if recording was saved in directory
rec_mm_per_sec	RA_INFO_REC_GEN_MM_SEC_D	mm per second (hint for displaing signal)

Table 6-6. List of info-names for device infos -- ra_dev_get_info

Info Name	Info Constant	Info Description
dev_hw_name	RA_INFO_REC_DEV_HW_NAME_C	name of the recording device
dev_hw_manufacturer	RA_INFO_REC_DEV_HW_MANUFACTURER_C	manufacturer of the recording device
dev_hw_serial_no	RA_INFO_REC_DEV_HW_SERIAL_NO_C	serial-number of the recording device
dev_hw_version	RA_INFO_REC_DEV_HW_VERSION_C	hardware-version of the recording device
dev_sw_name	RA_INFO_REC_DEV_SW_NAME_C	software-name of the recording device
dev_sw_manufacturer	RA_INFO_REC_DEV_SW_MANUFACTURER_C	manufacturer of the software of the recording device
dev_sw_serial_no	RA_INFO_REC_DEV_SW_SERIAL_NO_C	serial-number of the software of the recording device
dev_sw_version	RA_INFO_REC_DEV_SW_VERSION_C	version of the software of the recording device

Table 6-7. List of info-names for channel infos -- ra_ch_get_info

Info Name	Info Constant	Info Description
ch_name	RA_INFO_REC_CH_NAME_C	name of the channel
ch_desc	RA_INFO_REC_CH_DESC_C	description of the channel
ch_num_samples	RA_INFO_REC_CH_NUM_SAMPLES_L	number of samples recorded in the channel
ch_samplerate	RA_INFO_REC_CH_SAMPLERATE_D	samplerate used in the channel
ch_num_bits	RA_INFO_REC_CH_NUM_BITS_L	number of bits used for one sample
ch_amp_res	RA_INFO_REC_CH_AMP_RESOLUTION_D	value in ‘unit’ representing one unit
ch_unit	RA_INFO_REC_CH_UNIT_C	unit of the channel

Info Name	Info Constant	Info Description
ch_center_value	RA_INFO_REC_CH_CENTER_VALUE_D	value [unit] signal is centered
ch_center_sample	RA_INFO_REC_CH_CENTER_SAMPLE_L	center value as sample-value
ch_min_unit	RA_INFO_REC_CH_MIN_UNIT_D	minimum value in units
ch_max_unit	RA_INFO_REC_CH_MAX_UNIT_D	maximum value in units
ch_mm_per_unit	RA_INFO_REC_CH_MM_UNIT_D	mm per unit (hint for displaing signal)
ch_type	RA_INFO_REC_CH_TYPE_L	type of channel (see above RA_CH_TYPE_*)

Table 6-8. List of info-names for evaluation infos -- ra_eval_get_info

Info Name	Info Constant	Info Description
eval_name	RA_INFO_EVAL_NAME_C	name of evaluation (if any)
eval_comment	RA_INFO_EVAL_COMMENT_C	comment of evaluation
eval_add_timestamp	RA_INFO_EVAL_ADD_TS_C	timestamp when eval was added
eval_modify_timestamp	RA_INFO_EVAL_MODIFY_TS_C	timestamp when eval was last modified
eval_user	RA_INFO_EVAL_USER_C	user who added eval
eval_host	RA_INFO_EVAL_HOST_C	machine on which eval was added
eval_program	RA_INFO_EVAL_PROG_C	program which added eval
eval_is_original	RA_INFO_EVAL_ORIGINAL_L	evaluation done in recording system
eval_is_default	RA_INFO_EVAL_DEFAULT_L	evaluation which should be used
eval_num_sets	RA_INFO_EVAL_SET_NUM_L	number of event-sets
eval_num_properties	RA_INFO_EVAL_PROP_NUM_L	number of different event properties
eval_set_name	RA_INFO_EVSET_NAME_C	name of event-set
eval_set_desc	RA_INFO_EVSET_DESC_C	description of event-set
eval_set_type	RA_INFO_EVSET_TYPE_L	type of event-set (e.g. discrete, continues)
eval_set_num_events	RA_INFO_EVSET_EV_NUM_L	number of events in event-set

Table 6-9. List of info-names for event-property infos -- ra_prop_get_info

Info Name	Info Constant	Info Description

Info Name	Info Constant	Info Description
eval_prop_name	RA_INFO_EVPROP_NAME_C	name of event-property, must be unique
eval_prop_desc	RA_INFO_EVPROP_DESC_C	description of event property
eval_prop_value_type	RA_INFO_EVPROP_VALTYPE_L	type of the event-property values (short, long, double)
eval_prop_handle_template	RA_INFO_EVPROP_TEMPLATE_L	flag if property handles template numbers (0: no, 1: yes)
eval_prop_has_minmax	RA_INFO_EVPROP_HAS_MINMAX_L	flag if property has min-/max-values
eval_prop_min	RA_INFO_EVPROP_MIN_D	normal min-value of the property
eval_prop_max	RA_INFO_EVPROP_MAX_D	normal max-value of the property
eval_prop_unit	RA_INFO_EVPROP_UNIT_C	unit of the property

Table 6-10. List of info-names for template infos -- not yet implemented

Info Name	Info Constant	Info Description
templ_num	RA_INFO_TEMPL_NUM_L	number of templates
templ_ch_list	RA_INFO_TEMPL_NUM_CH_L	number of channels where templates where got
templ_ch	RA_INFO_TEMPL_CH_NUM_L	channel-number used
templ_ch_size	RA_INFO_TEMPL_CH_SIZE_L	number of data used for template in this channel

Table 6-11. List of info-names for single-template infos -- not yet implemented

Info Name	Info Constant	Info Description
templ_single_number	RA_INFO_TS_NUMBER_L	template number
templ_single_fiducial	RA_INFO_TS_FIDUCIAL_L	offset of fiducial point (-1 if none)
templ_single_type	RA_INFO_TS_TYPE_L	type of template (e.g. classification when ecg is used)

Table 6-12. List of info-names for plugin infos -- ra_plugin_get_info

Info Name	Info Constant	Info Description
pl_name	RA_INFO_PL_NAME_C	name of plugin, used for identification
pl_desc	RA_INFO_PL_DESC_C	description of plugin
pl_file	RA_INFO_PL_FILE_C	filename with path of the plugin
pl_use_it	RA_INFO_PL_USE_IT_L	flag if plugin shoul be used or not
pl_version	RA_INFO_PL_VERSION_C	version of plugin

Info Name	Info Constant	Info Description
pl_build_ts	RA_INFO_PL_BUILD_TS_C	build timestamp of plugin
pl_license	RA_INFO_PL_LICENSE_L	license used for plugin
pl_num_results	RA_INFO_PL_NUM_RESULTS_L	number of values returned from plugin

Table 6-13. List of info-names for plugin-results infos -- ra_proc_get_result_info

Info Name	Info Constant	Info Description
res_name	RA_INFO_PL_RES_NAME_C	short name of result (can be used as table header)
res_desc	RA_INFO_PL_RES_DESC_C	description of result
res_default	RA_INFO_PL_RES_DEFAULT_L	flag if result belongs to default-values (=1)

Chapter 7. Signals for Inter Plugin Communication

No special intro text.

select-event

Name

`select-event` — select evens with a specific value

Synopsis

```
select-event (event-name, event-value);
```

Arguments

`event-name - char-pointer`

name of an event-property

`event-value - double`

value of an event-property

Description

This signal will be emitted if a set of events (’event-name’) with a specific value (’event-value’) should be selected.

highlight-event

Name

`highlight-event` — highlight an event

Synopsis

```
highlight-event (event-set-name, event-number);
```

Arguments

event-set-name - char-pointer

name of an event-set

event-number - long

number of a single event

Description

This signal can be emitted from a plugin if an event was selected and this event should be emphasised in other view-plugins. The selected event has the number ’event-number’ and is in the event-set ’event-set-name’.

eval-change

Name

`eval-change` — evaluation has changed

Synopsis

```
eval-change (none);
```

Arguments

-

no argument

Description

This signal will be emitted if the default-evaluation has changed. The receiving plugins must re-read the evaluation data and correct their handling of the evaluation data (e.g. displaying a x/y-plot) if necessary.

start-event-change

Name

`start-event-change` — some event-change signals are coming

Synopsis

```
start-event-change (none);
```

Arguments

-

no argument

Description

This signal will be emitted if more than one event-change signal is coming. The plugin can decide if it will response to every single change or collect all changes and response to the "end-event-change" signal.

end-event-change

Name

`end-event-change` — close an start-event-change

Synopsis

```
end-event-change (none);
```

Arguments

-

no argument

Description

This signal ’closes’ an "start-event-change". If the plugin collected the event-changes, now it can handle the changes.

event-change

Name

`event-change` — event-value has changed

Synopsis

```
event-change (event-name, event-number);
```

Arguments

`event-name - char-pointer`

name of an event-property

`event-number - long`

number of a single event

Description

This signal will be emitted if the value of an event-property has changed. The receiving plugins must check if the event-property is handled by the plugin and can ignore it, if it is not handled. If it is handled, the plugin can update whatever is needed (e.g. replot event-property values). If ’event-number’ is -1, the whole event-property should be re-read.

region-change

Name

`region-change` —

Synopsis

```
region-change (rec-pos, rec-pos);
```

Arguments

`rec-pos - long`

position in recording stream

`rec-pos - long`

position in recording stream

Description

This signal will be emitted if something changed between two positions.

add-event

Name

`add-event` — an event was added

Synopsis

```
add-event (event-set-name, event-number);
```

Arguments

`event-set-name - char-pointer`

name of an event-set

`event-number - long`

number of a single event

Description

This signal will be emitted if a new event (with event-number ’event-number’) was added to event-set ’event-set-name’.

del-event-begin

Name

`del-event-begin` — start of an event-deletion

Synopsis

```
del-event-begin (event-set-name, event-number);
```

Arguments

event-set-name - char-pointer

name of an event-set

event-number - long

number of a single event

Description

This signal will be emitted before an event will be deleted. This enables plugins to get infos from the event before deletion. E.g. for templates it is necessary to know th template-number used by an event. The deleted event is event number ’event-number’ from the event-set ’event-set-name’.

del-event-end

Name

`del-event-end` — end of an event-deletion

Synopsis

```
del-event-end (event-set-name, event-number);
```

Arguments

event-set-name - char-pointer

name of an event-set

event-number - long

number of a single event

Description

This signal will be emitted after an event was deleted. The deleted event is event number ’event-number’ from the event-set ’event-set-name’.

template-delete

Name

`template-delete` — a template was deleted

Synopsis

```
template-delete (event-set-name, template-number);
```

Arguments

`event-set-name - char-pointer`

name of an event-set

`template-number - long`

number of a template

Description

This signal will be emitted after a template was deleted. The deleted template is template number ’template-number’ from the event-set ’event-set-name’.

template-add

Name

`template-add` — a template was added

Synopsis

```
template-add (event-set-name, template-number);
```

Arguments

event-set-name - char-pointer

name of an event-set

template-number - long

number of a template

Description

This signal will be emitted if a template was added. The added template is template number ’template-number’ from the event-set ’event-set-name’.

pos-change

Name

`pos-change` — set view to a new position

Synopsis

```
pos-change (rec-pos, max-x-scale);
```

Arguments

rec-pos - long

position in recording stream

max-x-scale - double

scaling factor for x-axis for channel with highest samplerate

Description

This signal will be emitted, if a view-plugin changed the position in the recording stream which is displayed. The new position is given by 'rec-pos' and is in sampleunits of the recording. View-plugins MUST handle this signal. (TODO: think about how to handle different recordings?)

gui-event-start

Name

`gui-event-start` — start of signal activated by GUI-event

Synopsis

```
gui-event-start (none);
```

Arguments

-

no argument

Description

This signal will indicate that the next signal(s) emitted is(are) released by a GUI-event (e.g. mouse click).

gui-event-end

Name

`gui-event-end` — end of signal activated by GUI-event

Synopsis

```
gui-event-end (none);
```

Arguments

-

no argument

Description

Indicating that the GUI-event is finished. (see 'gui-event-start').

x-resolution

Name

`x-resolution` — set new resolution of x-axis

Synopsis

```
x-resolution (x-res, gui-parent);
```

Arguments

x-res - double

resolution of the x-axis

gui-parent - long

id (e.g. the pointer) of the parent gui-element

Description

This signal will be emitted if the resolution of the x-axis of a view has changed. The new reslution is given by 'x-res'. Additionally the parent of the view is given by 'gui-parent'. Normally the view only change his x-resolution if the parent of the receiving view is the same as the sending view (sync. of splitted views).

y-resolution

Name

`y-resolution` — set new mm/unit of a channel

Synopsis

y-resolution (*mm-per-unit*, *channel*);

Arguments

mm-per-unit - double

mm/unit a channel

channel - long

channel-number of root-measurement

Description

This signal will be emitted if the mm/unit of a channel has changed. The new y-resolution is given by ’mm-per-unit’, the channel by ’channel’.

Chapter 8. libRASCH Plugins

No special intro text.

8.1. Access Plugins

ART

Name

ART — Handle signals recorded for ART-Study

Description

The plugin handles signals which are recorded for the ART-Study. To access the recorded data, the plugins for portilab and portapres are used.

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

cfs

Name

cfs — Handle signals saved using the CED Filing System (CFS)

Description

The plugin provides access to recordings saved in the CED Filing System (CFS) format. At the moment only equalspaced channels are supported.

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ctg-rasch

Name

`ctg-rasch` — Handle cardiotocogram signals exported from GMT system

Description

The plugin provides access to CTG's exported from the GMT system. The exported raw data (three files: fetal heart rate child 1, fetal heart rate child 2 and uterine contraction) is packed in a directory with the extension '.ctg'. Additionally some information about the mother and the recording is stored in the file 'ctg_info.txt'.

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ctg-sonicaid

Name

`ctg-sonicaid` — Handle cardiotocogram signals saved with the Oxford FetalCare system

Description

The plugin provides access to CTG's handled with the Oxford FetalCare system. The information about the Mother and the recording date and time is stored in a Access database-file. Therefore a text file similar to the one in the ctg-rasch plugin will be used to handle this information (the file is called 'ctg_info.txt').

Version

0.1.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

dadisp

Name

`dadisp` — Handle signals saved a DaDisp signal file format

Description

The plugin provides access to recordings saved in a DaDisp signal file format. Because I have not used the program DaDisp but have only some files, this plugin supports only this format (files with the extension 'dsp' and a ASCII header at the beginning).

Version

0.1.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

dasylab

Name

`dasylab` — Handle signals recorded using DasyLab

Description

The plugin provides access to recordings saved in the format used by the DasyLab-Systems.

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

edf/edf+

Name

`edf/edf+` — Handle signals recorded using European Data Format(+)

Description

This file contains source to handle XML annotations stored in EDF+ files. At the moment only heartbeat data is supported.

Remarks

If the EDF+ file is a non-contiguous signal, than each data-block is handled as a single session. (Recommendation from Jesus Olivan.)

Version

0.3.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ishne-holter

Name

`ishne-holter` — Handle signals saved using ISHNE standard output for Holter ECG

Description

The plugin provides access to recordings saved in the 'ISHNE standard output for Holter ECG' format.

Version

0.2.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

mit-bih

Name

`mit-bih` — Handle signals recorded using the MIT/BIH Data Format (Physiobank data sets)

Description

The plugin provides access to recordings saved in the format used for the MIT/BIH database(s) and the databases of provides by PhysioNet (www.physionet.org).

Version

0.3.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

portapres

Name

`portapres` — Handle signals recorded with Portapres-System

Description

The plugin provides access to blood-pressure data recorded with the Portapres system.

Remarks

Up to now only a small subset of the binary-file can be read. It is planned to add support to access the data exported with BeatScope.

Version

0.1.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

poly5/tms32

Name

`poly5/tms32` — Handle signals recorded using TMS ADC system and PortiLab

Description

The plugin provides access to data recorded with the TMS ADC-system and using the PortiLab software.

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

read-rri

Name

`read-rri` — Handle RR interval-files (nearly the final design)

Description

The plugin provides access to RR interval files. It now supports the first RR-interval format specific for libRASCH. The design of the format is not fixed and will be definitely changed in the not so distant future.

Version

0.1.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

8.2. Process Plugins

ap-morphology

Name

`ap-morphology` — get systolic and diastolyc values and positions of aterial pressure waves

Description

The plugin get systolic and diastolic values and positions from bloodpressure waves. Additionally it tries to find calibration-intervals (eg in the Portapres system it is possible to enable calibration during the measurement).

Options

Table 8-1. List of options for plugin ap-morphology

Name	Description	type
use_eval	use all positions in evaluation eh (function argument)	long
num_pos	number of positions where rr values and positions should be get	long
pos	positions	long array

Results

Table 8-2. List of results of plugin ap-morphology

Name	Description	type
IDX	index of event	long array
SYST	systolic blood pressures	double array
DIAS	diastolic blood pressures	double array
MEAN	mean blood pressures	double array
SYST_POS	positions of systolic measurements	long array
DIAS_POS	positions of diastolic measurements	long array
FLAGS	flags	long array
IBI	inter beat intervals	double array

Version

0.3.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

calibration

Name

`calibration` — measure calibration signals in a recording

Description

The plugin tries to detect and measure a calibration signal in a recording channel.

Options

Table 8-1. List of options for plugin calibration

Name	Description	type
rh	recording handle	long
ch	channel of the recording used to measure the calibration signal	long
start_pos	signal-position to start search for calibration signal [su]	long
end_pos	signal-position to stop search for calibration signal [su]	long
use_data	use the data given in 'data'	long
num_data	number of the data given in 'data'	long
data	perform calibration meausure using this data	double array
samplerate	samplerate used for the values in 'data'	double
type	type of calibration signal (see CALIB_TYPE_* in ra_calibration.h)	long
difference_lag	gap between samples used for difference signal [su]	long
min_calib_amp	percentage of maximum difference used as minimal calibration amplitude	double
segment_length	grid-width used to search for continous calibration signals [seconds]	double
min_level_length	minimum duration staying on the same level [seconds]	double

Name	Description	type
calib_cycle_length	length of one calibration cycle [seconds]	double
get_cycle_length	flag if the calibration cycle length should be measured	long
min_cycles	required minimum number of contiguous calibration cycles	long

Results

Table 8-2. List of results of plugin calibration

Name	Description	type
calib_height	calibration value	double
calib_high_value	high-level value of the calibration signal	double
calib_low_value	low-level value of the calibration signal	double
calib_width	width of the calibration signal (measured using 'calib_data')	double
calib_type	type of the calibration signal	long
calib_accuracy	accurcy of the calibration height [percent]	double
calib_out_of_range	flag if calibration cycles reached upper or lower value range	long
calib_data	data used to measure calibration (all valid segments combined)	double array
calib_cycle_pos	start- and end-position of valid calibration cycles	long array

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

dawes-redman

Name

`dawes-redman` — calculate FHR variations using the Dawes/Redman criteria

Description

The plugin calculates the fetal heart rate variations using the Dawes/Redman criteria. The estimation of the baseline of the fetal heart rate is done according the descriptions in the various papers from Dawes and Redman. For the digital filter, a Butterworth filter with order 4 and corner-frequency of 0.0017Hz (approx. 0.1 min-1)is used (using the 3.75sec epoch-values results in a samplerate of approx. 0.2667Hz). The filter was designed on the website of Tony Fisher, using the mkfilter cgi-script. This can be found at http://www-users.cs.york.ac.uk/~fisher/mkfilter/ .

Options

Table 8-1. List of options for plugin dawes-redman

Name	Description	type
use_start_end_pos	use values between start_pos and end_pos	long
start_pos	start-pos in sample-units	long
end_pos	end-pos in sample-units	long
rh	recording handle	long
ch_num	channel with the FHR	long
use_ignore_value	flag if there are values which are not valid	long
ignore_value	value which indicate not valid values	double
filter_maternal_pulse	run filter to find maternal pulse and remove these values	long
ignore_marked_regions	ignore regions which are marked in the annotations with the IGNORE flag	long
ignore_noise_regions	ignore regions which are marked in the annotations with the NOISE flag	long

Results

Table 8-2. List of results of plugin dawes-redman

Name	Description	type
FHR_EPOCHS	epochs of the fetal heart rate (averages over 3.75 seconds)	double array
FHR_BASELINE	baseline of the fetal heart rate	double array
SIGNAL_LOST	signal lost in percent	double
NUM_VALID_EPOCHS	number of valid epochs	long
BASAL_FHR	basal heart rate of the fetus	double
ACCEL_10	number of accelerations > 10bpm and <= 15bpm	long
POS_ACCEL_10	positions of the accelerations between 10 and 15bpm	long array
ACCEL_15	number of accelerations > 15bpm	long
POS_ACCEL_15	positions of the accelerations > 15bpm	long array
LOST_BEATS_20	decelerations < 20 lost beats	long
POS_LOST_BEATS_20	positions of the decelerations < 20	long array
LOST_BEATS_21_100	decelerations between 20 and 100 lost beats	long
POS_LOST_BEATS_21_100	postions of the decelerations between 20 and 100	long array
LOST_BEATS_101	decelerations above 100 lost beats	long
POS_LOST_BEATS_101	postions of the decelerations above 100	long array
MINUTE_RANGE	minute range for each minute in bpm	double array
MINUTE_RANGE_MS	miniute range for each minute in msec	double array
HIGH_VARIATIONS	number of minutes with high variations	long
MIN_HIGH_VARIATIONS	the minutes with the high variations	long array
LOW_VARIATIONS	number of minutes with low variations	long
MIN_LOW_VARIATIONS	the minutes with low variations	long array
LONG_TERM_VARIATIONS_MS	long term variations for all minutes in msec	double
LONG_TERM_VARIATIONS_BPM	long term variations for all minutes in bpm	double

Name	Description	type
LONG_TERM_VARIATIONS_HIGH_MS	long term variations for minutes with high variations in msec	double
LONG_TERM_VARIATIONS_HIGH_BPM	long term variations for minutes with high variations in bpm	double
SHORT_TERM_VARIATIONS	short term variations	double

Version

0.1.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

detect-ctg

Name

`detect-ctg` — perform detections in cardiotocograms

Description

The plugin performs detections of events in a cardiotocogram. At the moment only the position of the maximal uterine contraction will be searched.

Remarks

In the future, the plugin will be combined with the detect-simple plugin. For this step, the detect-simple plugin will be extended with options and the possibility to return the detection results not only by saving them in the evaluation file. But to get a faster result, the CTG-specific code will be first implemented here.

Options

Table 8-1. List of options for plugin detect-ctg

Name	Description	type
num_ch	number of channels set in 'ch'	long
ch	list of channels used for the event detection	long array

Results

Table 8-2. List of results of plugin detect-ctg

Name	Description	type
num	number of found events	long
pos	positions of the found events in sampleunits	long array
ch	bit-mask indicating the channels in which the event was found	long array
width	width of each event in sampleunits	long array
annot	annotations to the found events	long array

Version

0.1.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

detect-simple

Name

`detect-simple` — a simple beat-detection algorithm

Description

The plugin performs a simple beat detection in ecgs. The results will be saved in the libRASCH-evaluation-file of the measurement.

Options

Table 8-1. List of options for plugin detect-simple

Name	Description	type
num_ch	number of channels set in 'ch'	long
ch	list of channels used for the event detection	long array

Results

Table 8-2. List of results of plugin detect-simple

Name	Description	type
num	number of found events	long
pos	positions of the found events in sampleunits	long array
ch	bit-mask indicating the channels in which the event was found	long array
width	width of each event in sampleunits	long array
annot	annotations to the found events	long array

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ecg

Name

ecg — perform ecg specific actions after reading original evaluation or doing detect using libRASCH

Description

The plugin performs a finer classification of the qrs-complexes of an ecg, using from a previous evaluation only the information of the general annotation of a beat (coming from atrium, from ventricle, paced beat or artifact). Using this plugin allows to have in libRASCH a consistent sub-classification (eg premature normal beat) independent of the original evaluation system used.

Options

Table 8-1. List of options for plugin ecg

Name	Description	type
num_ref_rri	number rri used for calculation of reference-value	long
ref_normal_percent	percent within the previous rri will be uses as reference-value	long
normal_percent	max. tolerance for being a normal rri in percent	long
normal_pre_percent		long
normal_late_percent		long
max_abs_difference		long
vpc_pre_percent	max. percent of ref-value to be premature vpc	long
vpc_late_percent	min. percent of ref-value to be a ’real’ compensatory pause	long
min_interp_vpc_percent		long
max_interp_vpc_percent		long

Results

Table 8-2. List of results of plugin ecg

Name	Description	type
NUM_ALL	number of all QRS complexes	long

Name	Description	type
NUM_UNKNOWN	number of un-classified QRS complexes	long
IDX_UNKNOWN	event indeces of un-classified QRS complexes	long array
NUM_SINUS	number of sinus beats	long
IDX_SINUS	event indeces of sinus beats	long array
NUM_NORMAL	number of normal sinus-beats	long
IDX_NORMAL	event indeces of normal sinus-beats	long array
NUM_SVPC	number of SVPCs	long
IDX_SVPC	event indeces of SVPCs	long array
NUM_SINUS_VPC	number of premature sinus-beats which following beats come late (perhaps VENT?)	long
IDX_SINUS_VPC	event indeces of premature sinus-beats which following beats come late (perhaps VENT?)	long array
NUM_PAUSE	number of pause	long
IDX_PAUSE	event indeces of pause	long array
NUM_VENT	number of VPCs	long
IDX_VENT	event indeces of VPCs	long array
NUM_VENT_SINGLE	number of single VPCs	long
IDX_VENT_SINGLE	event indeces of single VPCs	long array
NUM_VENT_SINGLE_PREM	number of VPCs which are premature	long
IDX_VENT_SINGLE_PREM	event indeces of VPCs which are premature	long array
NUM_VENT_SINGLE_PREM_COMP	number of VPCs which are premature and have a compensatory pause	long
IDX_VENT_SINGLE_PREM_COMP	event indeces of VPCs which are premature and have a compensatory pause	long array
NUM_INTERP	number of interpolated VPCs	long
IDX_INTERP	event indeces of interpolated VPCs	long array
NUM_ESACPE	number of escape beats	long
IDX_ESCAPE	event indeces of escape beats	long array
NUM_PACED	number of paced beats	long
IDX_PACED	event indeces of paced beats	long array
NUM_ARTIFACT	number of artifacts	long

Name	Description	type
IDX_ARTIFACT	event indeces of artifacts	long array
NUM_ARTIFACT_TYPE1	number of automatic detected artifacts: rri is lower than the minimal allowed rri	long
IDX_ARTIFACT_TYPE1	event indeces of automatic detected artifacts: rri is lower than the minimal allowed rri	long array
NUM_ARTIFACT_TYPE2	number of automatic detected artifacts: perhaps detected T-wave	long
IDX_ARTIFACT_TYPE2	event indeces of automatic detected artifacts: perhaps detected T-wave	long array
NUM_ARTIFACT_TYPE3	number of automatic detected artifacts: perhaps overlooked beat after a VPC	long
IDX_ARTIFACT_TYPE3	event indeces of automatic detected artifacts: perhaps overlooked beat after a VPC	long array
MIN_HR	min. heart rate	double
IDX_MIN_HR	start event index of min. heart rate	long
MAX_HR	max. heart rate	double
IDX_MAX_HR	start event index of max. heart rate	long
MEAN_HR	mean heart rate	long
NUM_BRADY	number of bradycardia	long
IDX_BRADY	start event indeces of the bradicardia	long array
LEN_BRADY	length (#rri's) of the bradicardia	long array
HR_BRADY	heart rate of the bradicardia	double array
NUM_TACHY	number of tachycardia	long
IDX_TACHY	start event indeces of the tachycardia	long array
LEN_TACHY	length (#rri's) of the tachycardia	long array
HR_TACHY	heart rate of the tachycardia	double array
NUM_SSALVO	number of supraventricular salvos	long
IDX_SSALVO	start event indeces of supraventricular salvos	long array
LEN_SSALVO	length of supraventricular salvos	long
HR_SSALVO	heart rate of supraventricular salvos	double
NUM_SVT	number of supraventricular tachycardia	long

Name	Description	type
IDX_SVT	start event indeces of supraventricular tachycardia	long array
LEN_SVT	length of supraventricular tachycardia	long
HR_SVT	heart rate of supraventricular tachycardia	double
NUM_SSALVO_SVT	number of supraventricular salvos/SVT's	long
IDX_SSALVO_SVT	start event indeces of supraventricular salvos/SVT's	long array
LEN_SSALVO_SVT	length of supraventricular salvos/SVT's	long
HR_SSALVO_SVT	heart rate of supraventricular salvos/SVT's	double
NUM_IVR	number of ideoventricular rhythms	long
IDX_IVR	start event indeces of ideoventricular rhythms	long array
LEN_IVR	length (#rri's) of ideoventricular rhythms	long array
HR_IVR	heart rate of ideoventricular rhythms	double array
NUM_COUPLET	number of couplets	long
IDX_COUPLET	event indeces of couplets	long array
NUM_BIGEMINY	number of bigeminy	long
IDX_BIGEMINY	start event indeces of the bigeminy	long array
LEN_BIGEMINY	length (#rri's) of the bigeminy	long array
NUM_TRIGEMINY	number of trigeminy	long
IDX_TRIGEMINY	start event indeces of the trigeminy	long array
LEN_TRIGEMINY	length (#rri's) of the trigeminy	long array
NUM_SALVO	number of salvos	long
IDX_SALVO	start event indeces of the salvos	long array
LEN_SALVO	length (#rri's) of the salvos	long
HR_SALVO	heart rate of the salvos	double
NUM_VT	number of VTs	long
IDX_VT	start event indeces of VTs	long array
LEN_VT	length (#rri's) of the VTs	long
HR_VT	heart rate of the VTs	double
NUM_SALVO_VT	number of salvos/VTs	long

Name	Description	type
IDX_SALVO_VT	start event indeces of the salvos/VTs	long array
LEN_SALVO_VT	length (#rri's) of the salvos/VTs	long
HR_SALVO_VT	heart rate of the salvos/VTs	double

Version

0.3.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

fiducial-point

Name

`fiducial-point` — finds fiducial point of a QRS-complex

Description

The plugin finds the fiducial-points of the QRS-complexes in an ecg.

Options

Table 8-1. List of options for plugin fiducial-point

Name	Description	type
use_eval	use all positions in evaluation eh (function argument)	long
num_pos	number of positions where qrs-positions should be get	long
pos_are_eval_index	positions are index to events in eh	long
pos	postions	long array

Name	Description	type
ch	in which channel fiducial point should be searched	long

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

freq-analysis

Name

`freq-analysis` — perform frequency analysis

Description

Options

Table 8-1. List of options for plugin freq-analysis

Name	Description	type
use_events	!=0: use events, ==0: use signal raw data	long
evprop_name	use these event-property for frequency analysis	char *
evprop_pos_name	when using events, the position of the events is needed for time based analysis	char *
use_time	flag if freq. analysis should be done based on time(!=0) or based on the event(=0) (e.g. rr-interval)	long

Name	Description	type
use_start_end_event	use events from start_event to end_event	long
start_event	start-event (including this event)	long
end_event	end-event (excluding this event)	long
num_events	use the events listed in events	long
events	list of events to use	long array
num_ignore_events	number of events which should be ignored	long
ignore_events	list of events to ignore	long array
rh	recording handle	long
ch_num	use this channel for PRSA	long
use_ignore_value	flag if there are values which should not be used for freq. analysis	long
ignore_value	this value will be not used for freq. analysis (if use_ignore_value_1 is != 0)	double
use_start_end_pos	use raw-data/events between start_pos and end_pos	long
start_pos	start-pos in sample-units	long
end_pos	end-pos in sample-units	long
use_data	use the data given in 'data'	long
num_data	number of the data given in 'data'	long
data	perform frequency analysis using this data	double array
samplerate	samplerate to use when freq.analysis should be done time based	double
smooth_data	flag if input-data should be smoothed	long
smooth_method	at the moment only a boxcar-filter is available	char *
smooth_width	width of the smooth-filter	double
window	window function used	char *
remove_mean	remove the mean value before the frequency analysis	long
remove_trend	remove the trend (first order) before the frequency analysis	long
method	method used for the frequency analysis	char *
num_freq	number of frequency components	long

Name	Description	type
smooth_spec	flag if spectrum should be smoothed	long

Results

Table 8-2. List of results of plugin freq-analysis

Name	Description	type
FREQ_AXIS	frequency values (the x-axis)	double array
PSD	Power Spectrum Density	double array
REAL_PART	real part of the frequency spectrum	double array
IMG_PART	imaginary part of the frequency spectrum	double array

Version

0.2.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

hrv

Name

`hrv` — calculate heart rate variability (HRV) parameters

Description

The plugin calculates the heart rate variability (HRV) parameters. In the time domain the parameters, which are recommended from the Task Force for HRV, are calculated. The parameters from the frequency domain are calculated on the power-spectrum of the complete tachogram using FFT.

Options

Table 8-1. List of options for plugin hrv

Name	Description	type
use_start_end_pos	use events between start_pos and end_pos	long
start_pos	start-pos in sample-units	long
end_pos	end-pos in sample-units	long
use_start_end_event	use events from start_event to end_event	long
start_event	start-event	long
end_event	end-event	long
num_events	use the events listed in events	long
events	list of events to use	long array
use_user_freq_band	calculate also the power in the user-selected frequency band	long
user_freq_start	start of user-selected frequency band [Hz]	double
user_freq_end	end of user-selected frequency band [Hz]	double
skip_time_domain	do not calculate HRV in the time domain	long
skip_freq_domain	do not calculate HRV in the frequency domain	long

Results

Table 8-2. List of results of plugin hrv

Name	Description	type
SDNN	standard deviation of normal-to-normal intervals	double
HRVI	HRV-Index	double
SDANN	standard deviation of averaged normal-to-normal intervals	double
rmssd	root mean of squared sucsessive differences	double
pNN50		double
TP	total power	double
ULF	ultra low frequency power	double

Name	Description	type
VLF	very low frequency power of short-term recordings	double
LF	low frequency power	double
LF_NORM	normalised low frequency power	double
HF	high freuqency power	double
HF_NORM	normalised high frequency power	double
LF_HF_RATIO	LF/HF ratio	double
POWER_LAW	power law behavior	double
TACHO_INDEX	Event numbers used for HRV calculations	long array
USER_BAND	frequency power in a user-selected frequency band	double

Version

0.2.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

template

Name

`template` — combine events in templates

Description

The plugin group events which have similar raw-data (eg QRS-complexes which have a similar morphology).

Options

Table 8-1. List of options for plugin template

Name	Description	type
use_eval	use all positions in evaluation eh (function argument)	long
num_pos	number of positions where qrs-positions should be get	long
pos_are_eval_index	positions are index to events in eh	long
pos	postions	long array
pos_name	name of event-type which holds event positions	char *
templ_name	name of event-type which holds template numbers	char *
templ_corr	name of event-type which holds template correlation	char *
win_before	template-window-size before postion (in seconds)	double
win_after	template-window-size after position (in seconds)	double
accept	treshold which correlation values are accepted	double
slope_accept_low	min-value of slope of regression line	double
slope_accept_high	max-value of slope of regression line	double
num_ch	number of channels used for template creation	long
ch	list of channesl for which the template should be created	long array

Version

0.3.0

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

respiration

Name

`respiration` — calculates respiration parameters

Description

The plugin calculates parameters from a respiration signal. At the moment only chest measurement based respiration signals can be processed. At the moment only the mean of the chest measurement of the last inter beat interval is calculated. In the future other measures (e.g. freq. of respiration) will be implemented.

Options

Table 8-1. List of options for plugin respiration

Name	Description	type
use_eval	use values from evaluation	long
use_start_end_pos	use values between start_pos and end_pos	long
start_pos	start-pos in sample-units	long
end_pos	end-pos in sample-units	long
rh	recording handle	long
ch_num	channel with the respiration signal	long
eh	eval handle	long

Version

0.1.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

8.3. GUI/View Plugins

ch-select-dlg

Name

`ch-select-dlg` — dialog for selecting channels which will be used for processing

Description

Results

Table 8-1. List of results of plugin ch-select-dlg

Name	Description	type
CH	list of selected channels	long array

Version

0.2.1

Author(s)

eval-dlg

Name

`eval-dlg` — a dialog showing all evaluations for a measurement

Description

The plugin provides a dialog which shows all evaluation of a measurement which allows to delete evaluations and event-properties.

Version

0.1.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

plugin-info-dlg

Name

`plugin-info-dlg` — a dialog showing the list of all loaded plugins

Description

The plugin provides a dialog which shows information about the loaded plugins.

Version

0.1.3

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

sig-sel-dlg

Name

`sig-sel-dlg` — dialog to choose signales which can be handled with libRASCH

Description

The plugin provides a dialog which allows to choose a measurement. The dialog shows all supported measurements in a directory, which can be selected.

Options

Table 8-1. List of options for plugin sig-sel-dlg

Name	Description	type
initial_path	open dialog showing this path	char *

Results

Table 8-2. List of results of plugin sig-sel-dlg

Name	Description	type
SEL_FILE	selected file (incl. path)	char *

Version

0.2.1

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

simple-calc-dlg

Name

`simple-calc-dlg` — dialog to perform calculations using process plugins

Description

The plugin provides a dialog to perform calculations done process plugins. Only process plugins which return values (and not only store the results in the evaluation file like the beat-detect plugin) can be used (and only these can be selected).

Version

0.1.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

cont-ap-view

Name

`cont-ap-view` — view for continues arterial pressure recordings

Description

Version

0.1.5

Author(s)

ctg-view

Name

`ctg-view` — view for cardiotocograms

Description

The plugin allows to view ecg-stripes. The plugin use the ts-view plugin for showing the raw-data.

Version

0.1.4

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ecg-view

Name

`ecg-view` — view for ecg's

Description

The plugin allows to view ecg-stripes. The plugin use the ts-view plugin for showing the raw-data.

Version

0.1.5

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

plot-view

Name

`plot-view` — plot evaluation data

Description

The plugin plot evaluation data in a window and allows to control which data and in which format the data is plotted.

Version

0.1.3

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

rasch-view

Name

`rasch-view` — general view for libRASCH, handling all views

Description

This plugin is a general view-plugin, which tries to use for every supported type of signal (eg ecg, bloodpressure) the corresponding view-plugin. For ever channels which is not handled by a specific view-plugin, the default plugin is used (eg ts-view for time-series signals).

Remarks

Win32 specific: This view needs as parent a CMDIFrameWnd. The plugin creates a CMDIChildWnd and this will be returned. (The other view plugins don't create the CMDIChildWnd by themself, these plugins need a CMDIChildWnd as parent.)

Version

0.2.2

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

template-view

Name

`template-view` — view for templates

Description

The plugin shows the templates of an evaluation.

Version

0.1.4

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)

ts-view

Name

`ts-view` — view for time-series

Description

The plugin shows the raw-data for time-series signals. This plugin is the default-plugin for all time-series signals.

Version

0.4.5

Author(s)

Raphael Schneider (rasch@med1.med.tum.de)